# LCD16x02 and LM35 Sensor Interface with STM32

In this tutorial, we'll discuss how to interface the LM35 temperature sensor with an STM32F0 microcontroller. Using the ADC to get the analog output voltage of the sensor then converting it back to Celsius degrees, and finally display the result on an LCD 16x2.

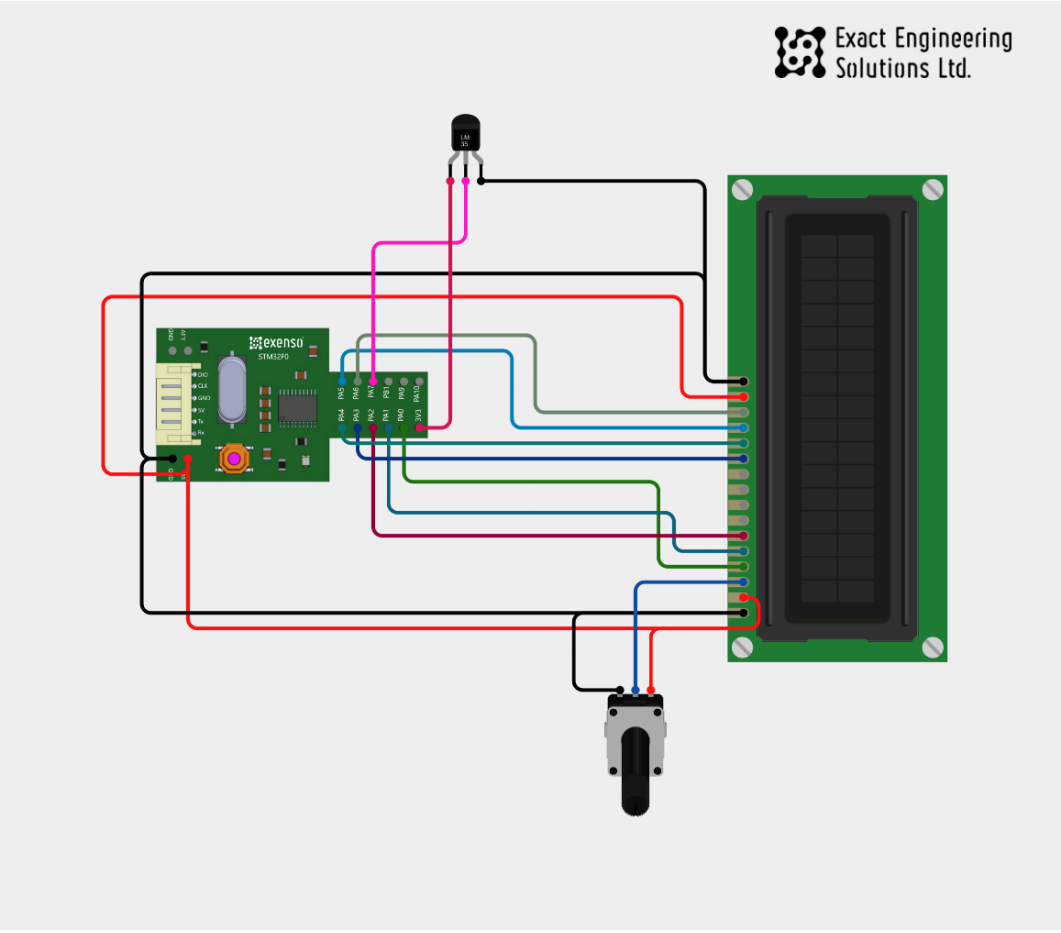
## Components Required

You will need the following components –

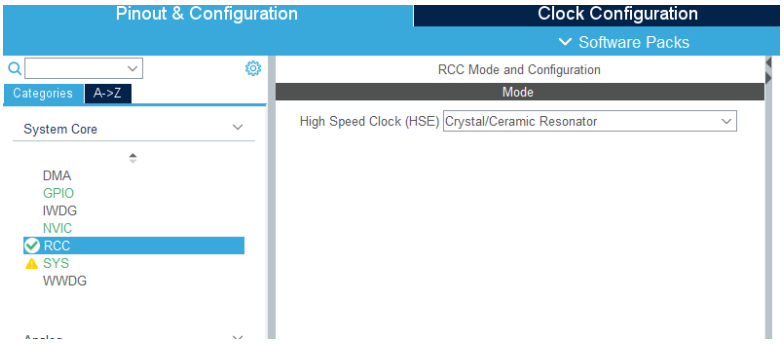
- 1 × Breadboard
- 1 × STM32F030F4P6
- 1× LCD 16x2
- 1× 10KΩ potentiometer
- 1x LM35 Temperature Sensor
- Some Jumper wire

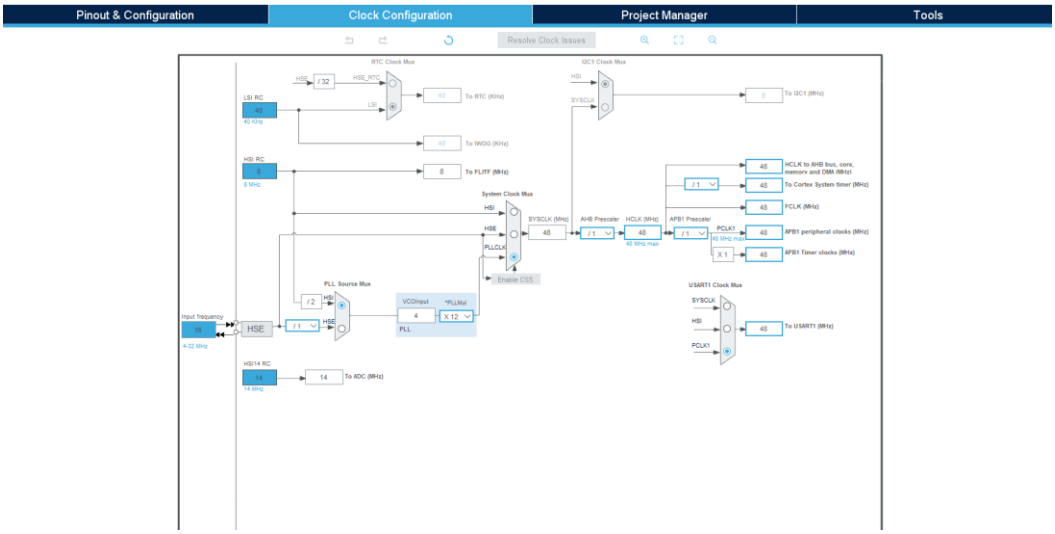
## Procedure

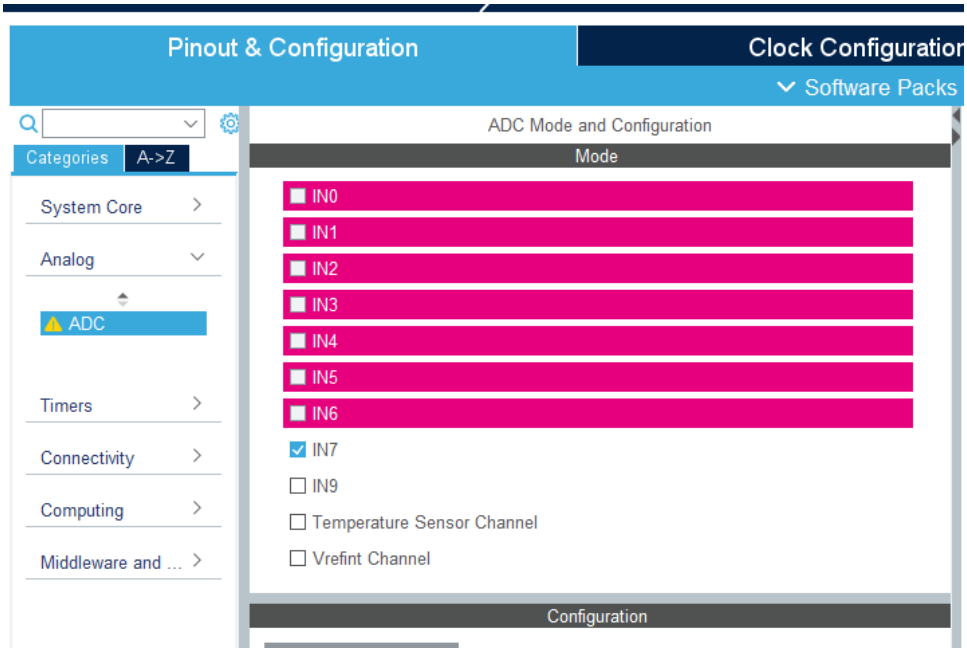
Follow the circuit diagram shown in the image given below.

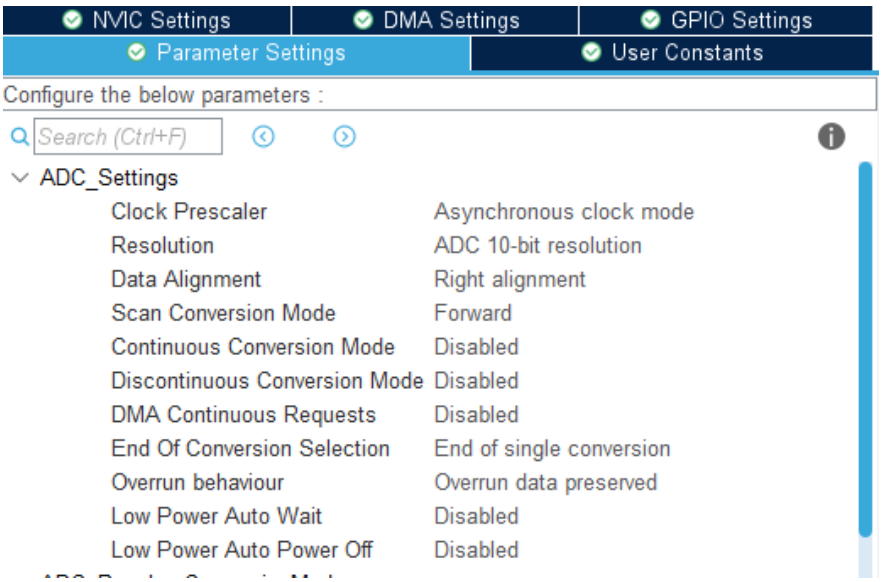


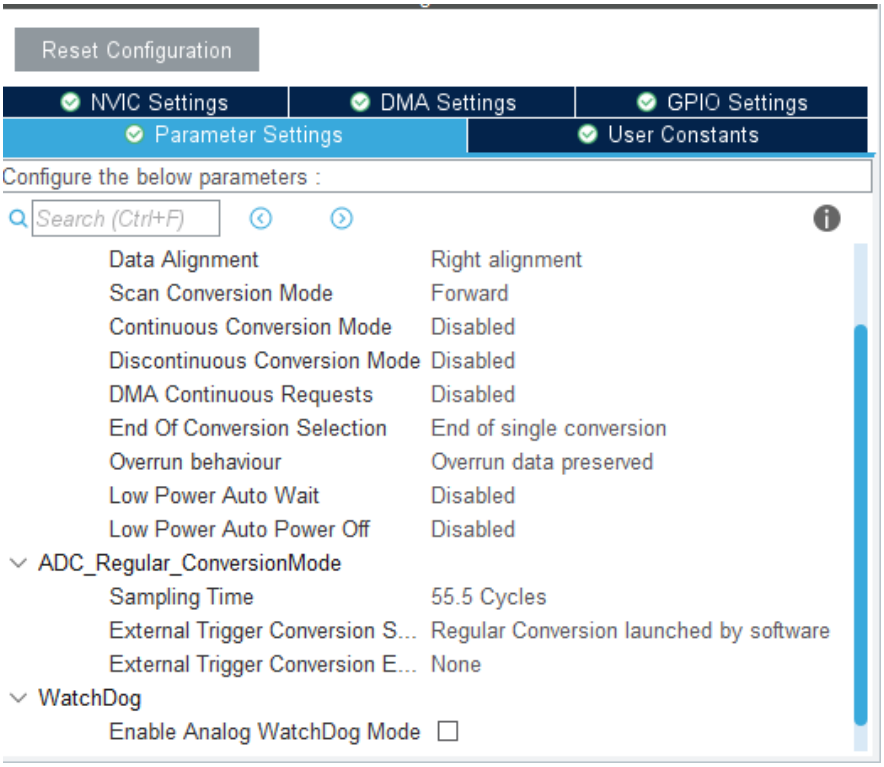
# STM32F0 Pin Configuration:

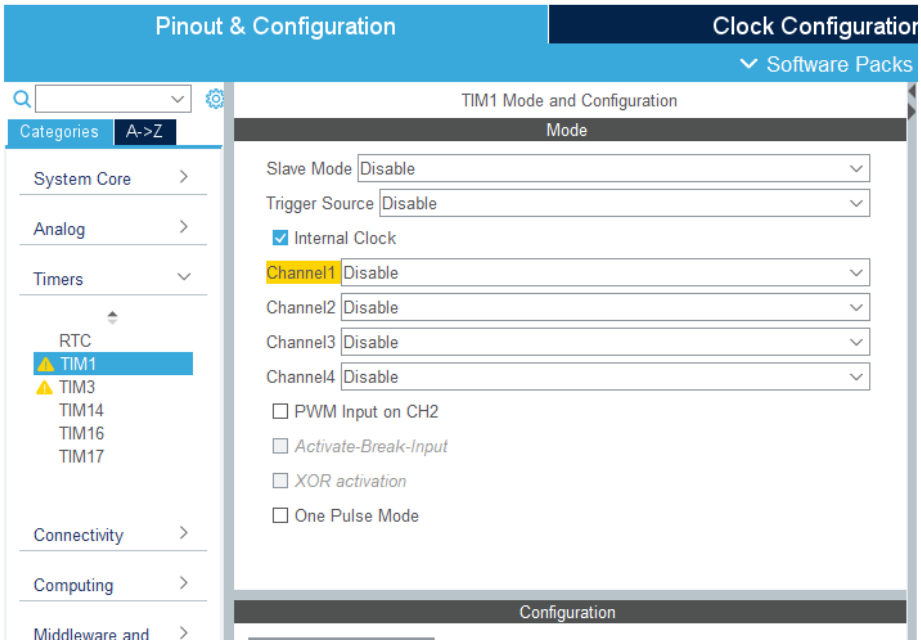


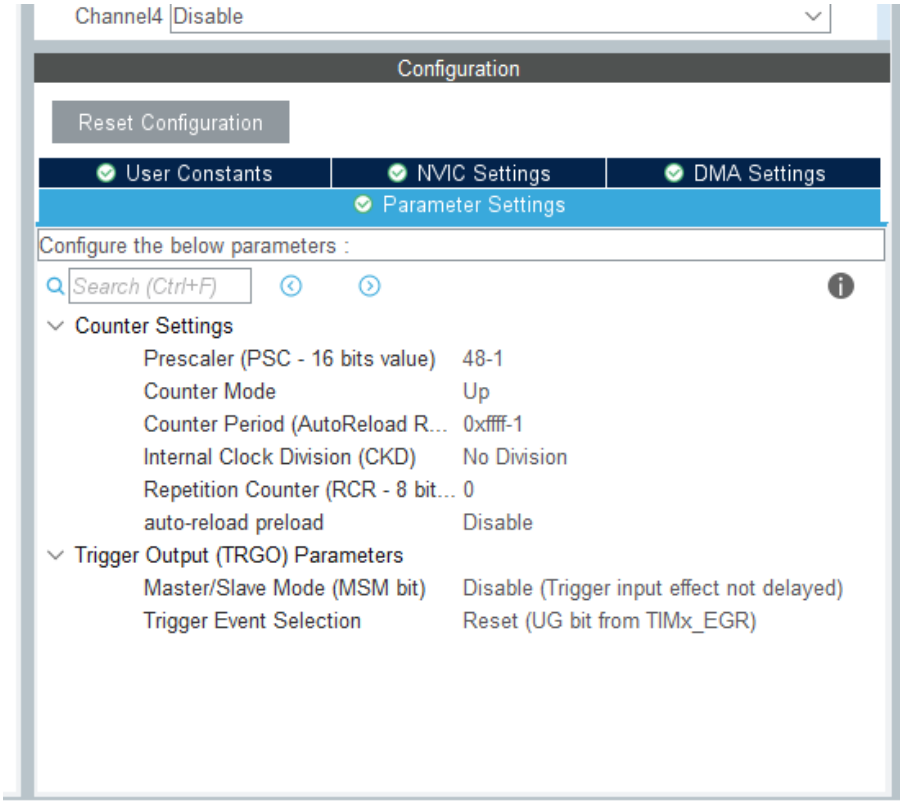



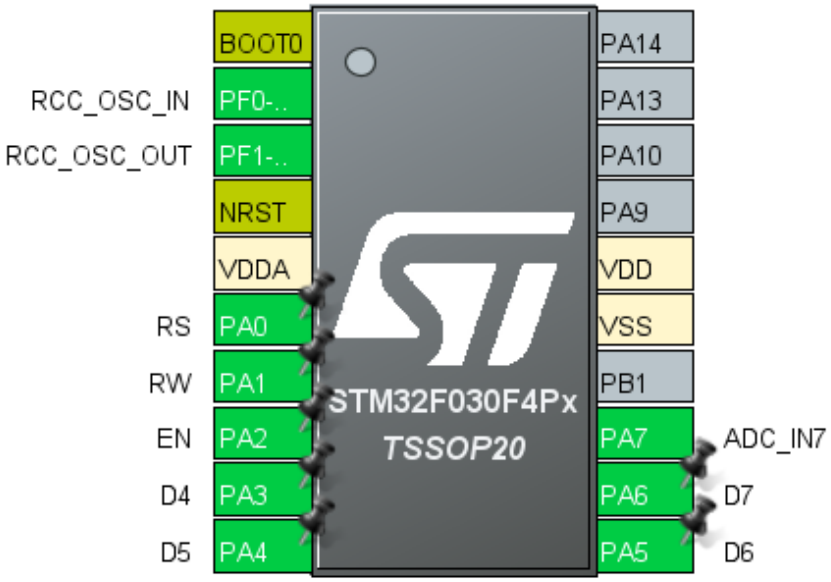

NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
ADC_Settings
Clock Prescaler
Asynchronous clock mode
Resolution
ADC 10-bit resolution
Data Alignment
Right alignment
Scan Conversion Mode
Forward
Continuous Conversion Mode
Disabled
Discontinuous Conversion Mode
Disabled
DMA Continuous Requests
Disabled
End Of Conversion Selection
End of single conversion
Overrun behaviour
Overrun data preserved
Low Power Auto Wait
Disabled
Low Power Auto Power Off
Disabled


Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
Data Alignment
Right alignment
Scan Conversion Mode
Forward
Continuous Conversion Mode
Disabled
Discontinuous Conversion Mode
Disabled
DMA Continuous Requests
Disabled
End Of Conversion Selection
End of single conversion
Overrun behaviour
Overrun data preserved
Low Power Auto Wait
Disabled
Low Power Auto Power Off
Disabled
ADC_Regular_ConversionMode
Sampling Time
55.5 Cycles
External Trigger Conversion S...
Regular Conversion launched by software
External Trigger Conversion E...
None
WatchDog
Enable Analog WatchDog Mode


Pinout & Configuration
Clock Configuration
Software Packs
Categories
A->Z
System Core
Analog
Timers
RTC
TIM1
TIM3
TIM14
TIM16
TIM17
Connectivity
Computing
Middleware and
TIM1 Mode and Configuration
Mode
Slave Mode Disable
Trigger Source Disable
Internal Clock
Channel1 Disable
Channel2 Disable
Channel3 Disable
Channel4 Disable
PWM Input on CH2
Activate-Break-Input
XOR activation
One Pulse Mode
Configuration


Channel4 Disable
Configuration
Reset Configuration
User Constants
NVIC Settings
DMA Settings
Parameter Settings
Configure the below parameters :
Search (Ctrl+F)
Counter Settings
Prescaler (PSC - 16 bits value) 48-1
Counter Mode Up
Counter Period (AutoReload R... 0xffff-1
Internal Clock Division (CKD) No Division
Repetition Counter (RCR - 8 bit... 0
auto-reload preload Disable
Trigger Output (TRGO) Parameters
Master/Slave Mode (MSM bit) Disable (Trigger input effect not delayed)
Trigger Event Selection Reset (UG bit from TIMx_EGR)

PA0 To PA6 Pin set as OUTPUT



## Code

```
#include "main.h"
#include "LCD1602.h"

float LM35_Read(uint16_t ADCvalue , float Ref_voltage, uint16_t bit_Resulation){

       float voltage  = ( ADCvalue * Ref_voltage) / bit_Resulation;
       float Temp = voltage / 0.01;
       return Temp;
}

ADC_HandleTypeDef hadc;
TIM_HandleTypeDef htim1;

void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM1_Init(void);
static void MX_ADC_Init(void);

float temp = 0.0;     // Init int value
uint16_t adcvalue = 0;

int main(void)
{
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_TIM1_Init();
  MX_ADC_Init();

  HAL_TIM_Base_Start(&htim1);  // Timer On and init this line
  lcd_init();                //LCD init
  lcd_xy(0, 4);   // set curser postion row 0 and colum 4
  LCD_String("LM35 Temp");  // Print String

  while (1)
  {
     // ADC conversion
       HAL_ADC_Start(&hadc);
       HAL_ADC_PollForConversion(&hadc,200);
       adcvalue = HAL_ADC_GetValue(&hadc);

      // Ref voltage 3.3V and 10bit = 2^10 = 1023
       temp = LM35_Read(adcvalue, 3.3, 1023);

       lcd_xy(1, 5);
       LCD_floatValue(temp, 2);  // print Float value and after dot show 2digit
       LCD_String("\337C");  // for degree celcus
       HAL_Delay(500);

  }

}
```

**Output:**